Partial Reconfiguration IP Core

2015.05.04

UG-PARTRECON

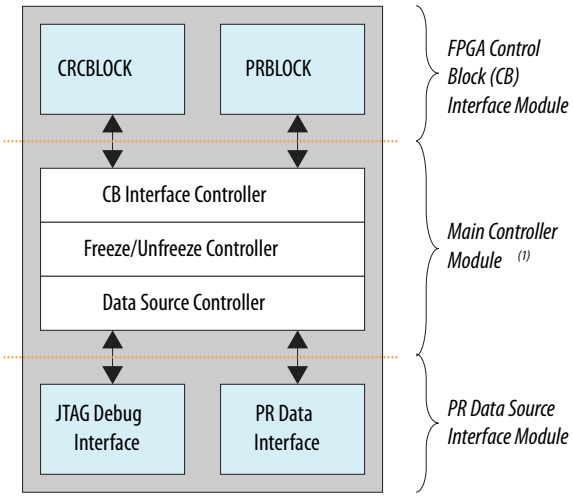
Subscribe Send Feedback

Partial reconfiguration (PR) is fully supported in the Stratix® V device family, which offers you the ability to reconfigure part of the design's core logic such as LABs, MLABs, DSP, and RAM, while the remainder of the design continues running. The PR IP core can be implemented through the Qsys Interface, or via the Quartus II® IP Catalog.

Partial reconfiguration is performed through either an internal host residing in the core logic or as an external host via dedicated PR pins. The advantage of the internal host is that you can store all the logic needed for PR on the device, without the need for external devices.

Figure 1: PR IP core Components

When you instantiate the PR IP core, the Main Controller module which includes the Control Block Interface Controller, Freeze/Unfreeze Controller, and the Data Source Controller are all instantiated. A Data Source Interface module provides you with a JTAG Debug Interface and PR Data Interface. If you choose to use the PR IP core as an internal host, it automatically instantiates the corresponding `crcblock` and `prblock` WYSIWYG atom primitives.



Note:
1. The main controller module handles all the handshaking signals of the CB interface and processes the incoming data, as needed, before sending to the PRBLOCK. It also handles the freeze/un-freeze PR interface.

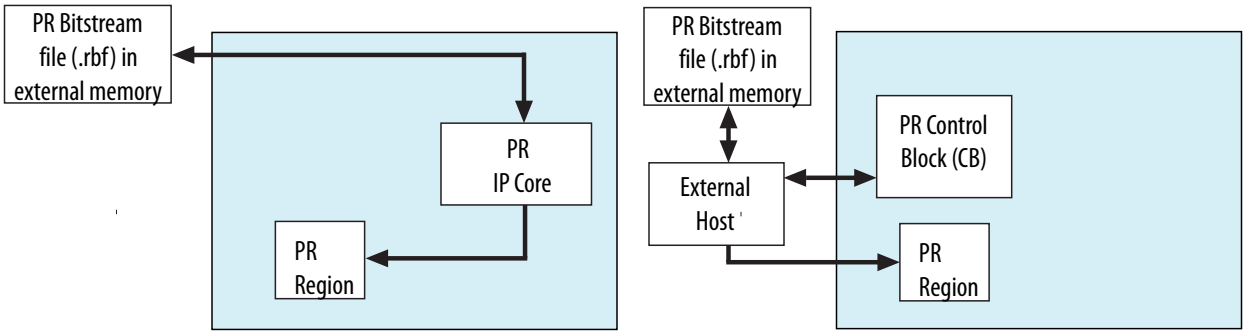
If it is used as external host (placed in another FPGA or CPLD), the PR IP core provides the `crcblock` and `prblock` WYSIWYG atom primitive as interface ports so that you can connect to the dedicated PR pins and `CRC_ERROR` pin on the target FPGA undergoing partial reconfiguration.

© 2015 Altera Corporation. All rights reserved. ALTERA, ARRIA, CYCLONE, ENPIRION, MAX, MEGACORE, NIOS, QUARTUS and STRATIX words and logos are trademarks of Altera Corporation and registered in the U.S. Patent and Trademark Office and in other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders as described at www.altera.com/common/legal.html. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

ISO 9001:2008 Registered

Figure 2: Managing Partial Reconfiguration with an Internal or External Host

The figure shows how these blocks should be connected to the PR control block (CB). In your system, you will have either the external host or the internal host, but not both. During PR, the PR Control Block (CB) is in Passive Parallel x16 programming mode.



Related Information

- **FPGA Control Block Interface** on page 15
- **Control Block Interface Controller** on page 17
- **Freeze and Unfreeze Controls** on page 18
- **Data Source Controller** on page 18
- **Standard Partial Reconfiguration Data Interface** on page 18
- **JTAG Debug Mode for Partial Reconfiguration** on page 19

Instantiating the Partial Reconfiguration IP Core in the Qsys Interface

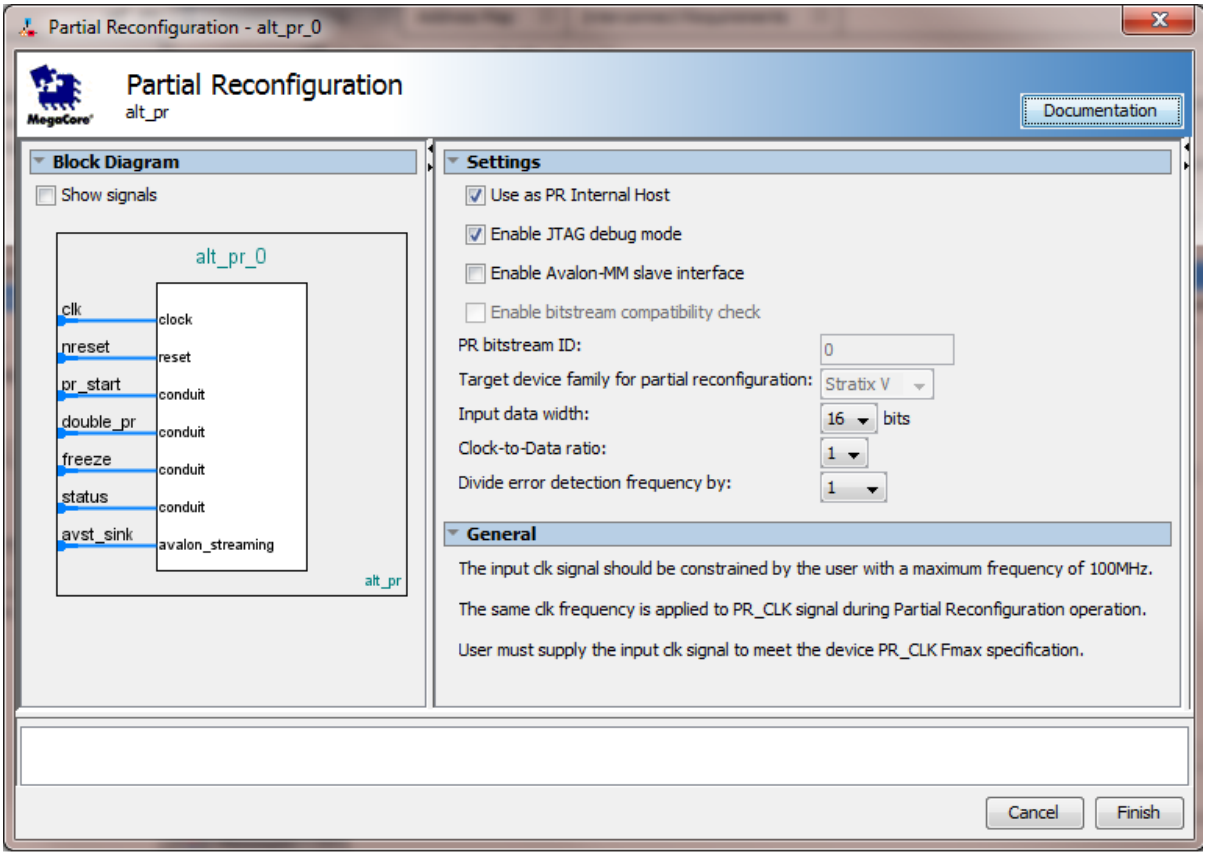
Partial Reconfiguration(PR) is available as a Qsys component through the Qsys interface.You can choose to instantiate the core as an internal host or an external host.

When instantiated with Qsys, PR is configured as a **Conduit** interface, or by enabling the **Avalon Memory Map Slave** interface. If you use Qsys and want PR included as component, you must instantiate the PR IP core in the Qsys interface.

To instantiate the PR IP core with Qsys:

1. Click **Tools** > **Qsys**
2. In the **Qsys** interface **IP Catalog** expand **Basic Functions** > **Configuration and Programming** and select **Partial Reconfiguration**.
3. Configure your IP core variation using the settings appropriate to your design.

Figure 3: Partial Reconfiguration IP Core in the Qsys Interface



4. Turn on **Enable Avalon-MM slave interface** to use the Avalon Memory Map Slave interface rather than the Conduit interface.
5. Click **Finish**.

Related Information

- **Instantiating the Partial Reconfiguration IP Core in the Quartus II IP Catalog** on page 3
- **Partial Reconfiguration IP Core Parameters** on page 6
- **Creating a System With Qsys**

Instantiating the Partial Reconfiguration IP Core in the Quartus II IP Catalog

Partial Reconfiguration(PR) is available from the **IP Catalog**.You can choose to instantiate the core as an internal host or an external host.

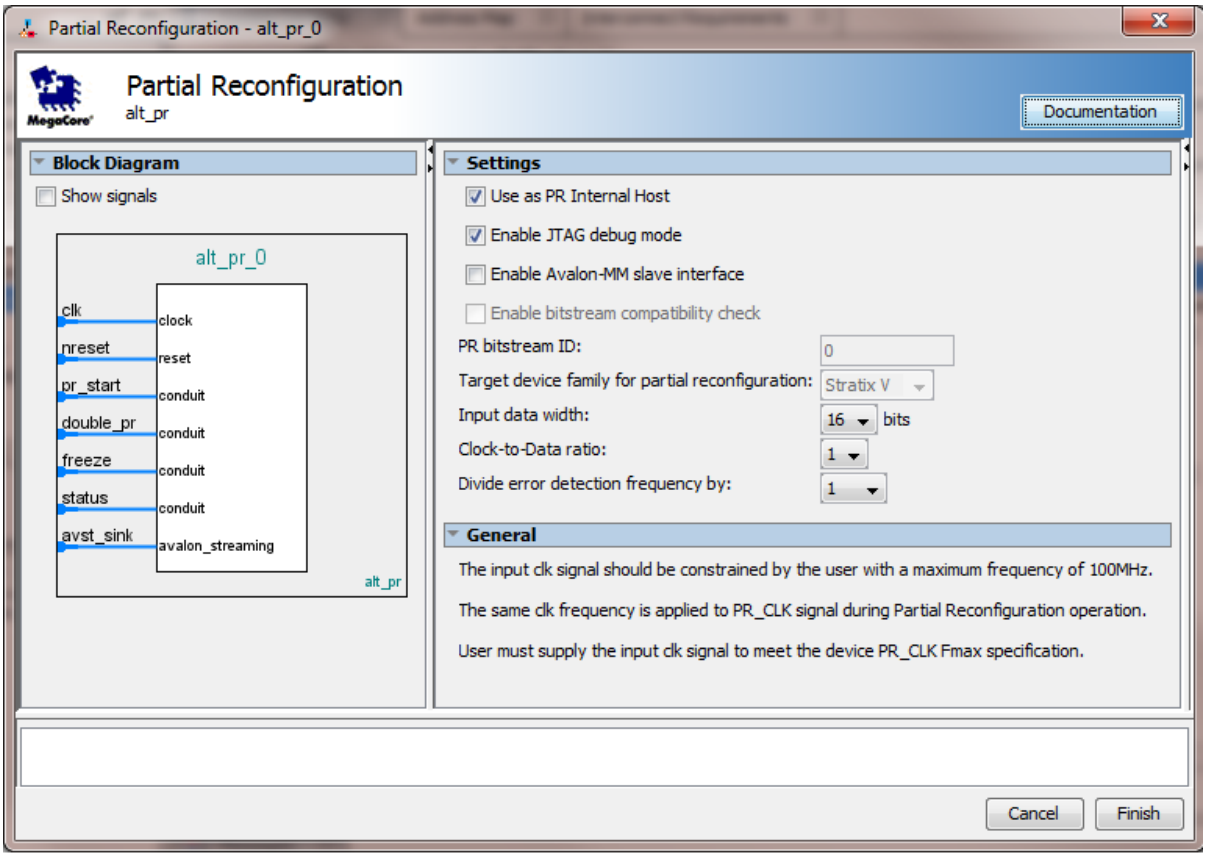
If you are not using PR as a component of the **Qsys** interface, then you can instantiate PR with the Quartus II IP Catalog.

The PR IP core can be instantiated as the internal host for Stratix V devices. When internal host is specified, both `prblock` and `crcblock` WYSIWYG atom primitives are auto-instantiated as part of the

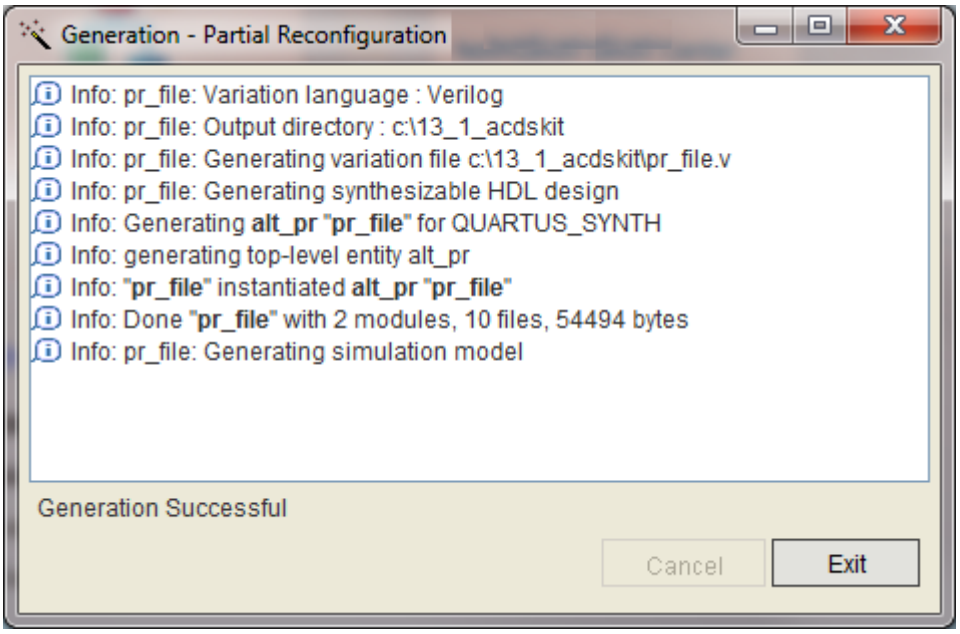
design. You can instantiate the PR IP core as the external host on any supported Altera devices as specified in the user selectable device family list.

1. Click **Tools** > **IP Catalog**.
2. Expand **Installed IP** > **Library** > **Basic Functions** > **Configuration and Programming** and select **Partial Reconfiguration**.
3. In the **Save IP Variation** dialog box, name your partial reconfiguration IP variation. Choose whether to use Verilog or VHDL. Click **OK** to save your variation.
4. Configure your IP core variation using the s appropriate to your design.

Figure 4: Partial Reconfiguration IP Core in the IP Catalog



5. Turn on **Enable Avalon-MM slave interface** to use the Avalon Memory Map Slave interface rather than the Conduit interface.
6. Click **Finish**.
   The IP Catalog instantiates your IP core variation and displays a completion dialog box.


Generation - Partial Reconfiguration
Info: pr_file: Variation language : Verilog
Info: pr_file: Output directory : c:\13_1_acdskit
Info: pr_file: Generating variation file c:\13_1_acdskit\pr_file.v
Info: pr_file: Generating synthesizable HDL design
Info: Generating alt_pr "pr_file" for QUARTUS_SYNTH
Info: generating top-level entity alt_pr
Info: "pr_file" instantiated alt_pr "pr_file"
Info: Done "pr_file" with 2 modules, 10 files, 54494 bytes
Info: pr_file: Generating simulation model
Generation Successful
Cancel
Exit

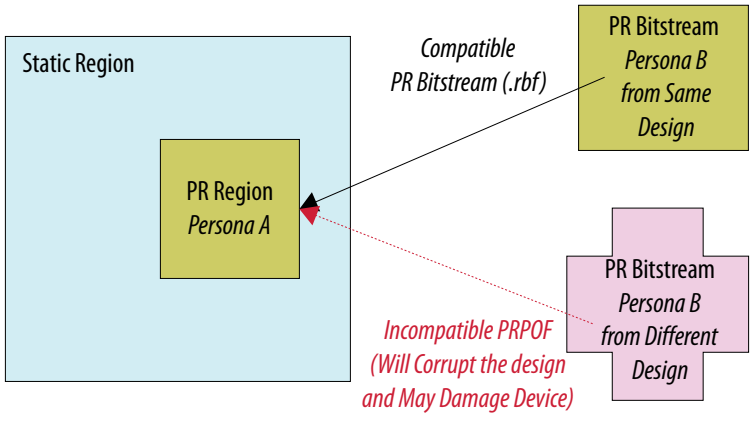
7. Click **Exit**.

Related Information

- **Instantiating the Partial Reconfiguration IP Core in the Qsys Interface** on page 2
- **Partial Reconfiguration IP Core Parameters** on page 6

Bitstream Compatibility Check

Turn on the **Enable bitstream compatibility check** when instantiaing the PR IP core from either Qsys or the IP Catalog to have the Quartus II software verify the partial reconfiguration PR Bitstream file (**.rbf**). If an incompatible bitstream is detected, the PR operation aborts and the `status` output reports an error.



This prevents you from accidentally corrupting the static region of your design with a bitstream from an incompatible **.rbf** and risking damage to the chip being programmed.

When **Enable bitstream compatibility check** is turned on, the PR IP core creates a **PR bitstream ID** and displays it in the configuration dialog box.

Related Information

- **Partial Reconfiguration IP Core Parameters** on page 6
- **Partial Reconfiguration IP Core Ports** on page 8

Partial Reconfiguration IP Core Parameters

IP Core Option	Value	Default	Description
Use as PR Internal Host	On or Off	On	Turn on this option to use the PR IP core as an internal host. Both `prblock` and `crcblock` WYSIWYG atom primitives are auto-instantiated as part of your design. Disable this option to use the PR IP core as an external host. You must connect additional interface signals to the dedicated PR pins or the external `prblock` and `crcblock` WYSIWYG atom primitives interface signals if the PR IP core is used as an external host.
Enable JTAG debug mode	On or Off	On	Turn on this option to access the PR IP core with the Programmer to perform partial reconfiguration.
Enable Avalon-MM slave interface	On or Off	Off	Turn on this option to use the Avalon Memory Map slave interface
Enable bitstream compatibility check	On or Off	Off	Turn on this option to check the bitstream compatibility during PR operations for External Host. The bitstream compatibility check feature is always enabled for PR Internal Host. The PR bitstream ID value must be specified if this option is enabled for PR External Host.

IP Core Option	Value	Default	Description
PR bitstream ID	-2147483648 to 2147483647	0	Specifies a signed 32-bit integer value of the PR bitstream ID for External Host. This value must match the PR bitstream ID generated during compilation for the target PR design. The PR bitstream ID value of the target PR design can be found in the Assembler compilation report (.asm.rpt).
Input Data Width	1, 2, 4, 8, 16, or 32	16	Specifies the data width in bits. This option affects the `data[]` bus width.
Target device familiy for partial reconfiguration	"Arria V", "Arria V GZ", "Cyclone V", "Stratix V"	"Stratix V"	Select the target device family for partial reconfiguration when the PR megafunction is used as External Host. **Note:** This option is ignored for PR Internal Host.
Clock-to-Data ratio	1, 2, or 4	1	Specifies the ratio between PR clock and PR data. Select '1' for plain PR data, '2' for encrypted PR data, or '4' for compressed PR data (with or without encryption)
Divide error detection frequency by	1, 2, 4, 8, 16, 32, 64, 128, or 256	1	Only available when the IP core is used as an Internal Host where the `crcblock` WYSIWYG atom primitive is auto-instantiated as part of the design. Specifies the divide value of the internal clock, which determines the frequency of the error detection CRC. The divide value must be a power of two. Refer to the device handbook to find the frequency of the internal clock for the selected device.

Related Information

- **Using the Avalon Memory Mapped Slave Interface** on page 13
- **Avalon Memory Map Slave Interface Read and Write Transfer Timing** on page 17
  For more information on the timing specification for the Avalon Memory Mapped Slave interface.

Partial Reconfiguration IP Core Ports

I/O Port List for PR IP Core

Table 1: Clock/Reset Ports

These options are always available.

Port Name	Width	Direction	Function
`nreset`	1	Input	Asynchronous reset for the PR IP core. Set high to enable partial reconfiguration. Set low to prevent partial reconfiguration and reset the state machine in the PR IP core.
`clk`	1	Input	User input clock to the PR IP core. This signal is ignored during JTAG debug operations.

Table 2: Conduit Interface

This option is always available.

Port Name	Width	Direction	Function
`freeze`	1	Output	Active high signal used to freeze the PR interface signals of the region undergoing partial reconfiguration. De-assertion of this signal indicates the end of PR operation.

Table 3: Conduit Interface

These options are available when **Enable Avalon-MM slave interface** parameter is turned **Off**.

Port Name	Width	Direction	Function
pr_start	**1**	Input	A signal arriving at this port asserted high initiates a PR event. You must assert this signal high for a minimum of one clock cycle and de-assert it low prior to the end of the PR operation so that the PR IP core is ready to accept the next pr_start trigger event when the freeze signal is low. This signal is ignored during JTAG debug operations.
data[]	**1, 2, 4, 8, 16, or 32**	Input	Selectable input PR data bus width, either x1, x2, x4, x8, x16, or x32. Once a PR event is triggered, it is synchronous with the rising edge of the clk signal whenever the data_valid signal is high and the data_read signal is high. This signal is ignored during JTAG debug operations.
data_valid	**1**	Input	A signal arriving at this port asserted high indicates the data[] port contains valid data. This signal is ignored during JTAG debug operations.
data_ready	**1**	Output	A signal arriving at this port asserted high indicates the PR IP core is ready to read the valid data on the data[] port whenever the data_valid signal is asserted high. The data sender must stop sending valid data if this port is low. This signal deasserted low during JTAG debug operations.

Port Name	Width	Direction	Function
status[2..0]	3	Output	A 3-bit error output used to indicate the status of PR event. Once an error is detected (PR_ERROR, CRC_ERROR, or Incompatible bitstream error), this signal is latched high and only get reset at the beginning of the next PR event, when pr_start is high and freeze is low. For example: 3'b000 – power-up or nreset asserted 3'b001 – PR_ERROR was triggered 3'b010 – CRC_ERROR was triggered 3'b011 – Incompatible bitstream error detected 3'b100 – PR operation in progress 3'b101 – PR operation passed 3'b110 – Reserved 3'b111 – Reserved

Port Name	Width	Direction	Function
double_pr	1	Input	When the pr_start signal is triggered until the de-assertion of freeze signal, a signal asserted high on this port indicates the PR event requires double PR cycle. A low signal on this port indicates a single PR cycle event. If your design requires the use of double PR because you have initialized RAM in the PR region, you must assert the double_pr input signal high so that the controller can handle double PR properly. You must assert this signal high if the PR bitstream (**.rbf**) is generated with the **Write memory contents** option turned on. Failure to do so causes a PR_ERROR assertion during partial reconfiguration. This signal is ignored during JTAG debug operations.

Table 4: Avalon-MM Slave Interface

These options are available when **Enable Avalon-MM Slave Interface** parameter is turned **On**.

Port Name	Width	Direction	Function
avmm_slave_address	1	Input	Avalon-MM address bus. The address bus is in the unit of Word addressing. Refer to the Qsys Component section for more details on the address mapping. This signal is ignored during JTAG debug operations.
avmm_slave_read	1	Input	Avalon-MM read control. This signal is ignored during JTAG debug operations.
avmm_slave_readdata	**16** or **32**	Output	Avalon-MM read data bus. This signal is ignored during JTAG debug operations.

Port Name	Width	Direction	Function
avmm_slave_write	**1**	Input	Avalon-MM write control. This signal is ignored during JTAG debug operations.
avmm_slave_writedata	**16** or **32**	Input	Avalon-MM write data bus. This signal is ignored during JTAG debug operations.
avmm_slave_waitrequest	**1**	Output	Asserted to indicate that the IP is busy and it is unable to respond to a read or write request. This signal is pulled high during JTAG debug operations.

Table 5: External Host Interface

These options are available when **Use as PR Internal Host** parameter is turned **Off**.

Port Name	Width	Direction	Function
crc_error_pin	**1**	Input	Available when you use the PR IP core as an External Host. Connect this port to the dedicated CRC_ERROR pin of the FPGA undergoing partial reconfiguration, or connect directly to the crcblock WYSIWYG atom primitive.
pr_ready_pin	**1**	Input	Available when you use the IP core as an External Host. Connect this port to the dedicated PR_READY pin of the FPGA undergoing partial reconfiguration, or connect directly to the prblock WYSIWYG atom primitive.
pr_error_pin	**1**	Input	Available when the IP is used as the External Host. Connect this port to the dedicated PR_ERROR pin of the FPGA undergoing partial reconfiguration, or connect directly to the prblock WYSIWYG atom primitive.

Port Name	Width	Direction	Function
pr_done_pin	1	Input	Available when the PR IP core is used as the External Host. Connect this port to the dedicated PR_DONE pin of the FPGA undergoing partial reconfiguration, or connect directly to the prblock WYSIWYG atom primitive.
pr_request_pin	1	Output	Available when the IP is used as the External Host. Connect this port to the dedicated PR_REQUEST pin of the FPGA undergoing partial reconfiguration, or connect directly to the prblock WYSIWYG atom primitive.
pr_clk_pin	1	Output	Available when the IP is used as the External Host. Connect this port to the dedicated DCLK of the FPGA undergoing partial reconfiguration, or connect directly to the prblock WYSIWYG atom primitive.
pr_data_pin[15..0]	16	Output	Available when the IP is used as the External Host. Connect this port to the dedicated DATA[15..0] pins of the FPGA undergoing partial reconfiguration, or connect directly to the prblock WYSIWYG atom primitive.

Related Information

Avalon Memory Map Slave Interface Data/CSR Memory Map on page 14

Using the Avalon Memory Mapped Slave Interface

Perform partial reconfiguration through the Avalon Memory Mapped Slave interface by following these steps:

1. Avalon Memory Mapped Master component writes `16'h0001` (or `16'h0003` if the design requires double PR) to this IP address offset 0x1 to trigger PR operation.
2. Avalon Memory Mapped Master component writes PR bitstream (data width in x16) to this IP address offset 0x0 until all the PR bitstream is written.
3. Avalon Memory Mapped Master component reads the data from this IP address offset 0x1 to check the `status[2:0]` value. Optionally, Avalon-MM Master component can read the `status[2:0]` of this IP during PR operation to understand if any early failure was detected, for example, `PR_ERROR`.

Related Information

- **Partial Reconfiguration IP Core Parameters** on page 6
- **Avalon Memory Map Slave Interface Read and Write Transfer Timing** on page 17
  For more information on the timing specification for the Avalon Memory Mapped Slave interface.

Avalon Memory Map Slave Interface Data/CSR Memory Map

Table 6: Data/CSR Memory Map Format

Name	Address Offset	Width	Access	Description
`PR_DATA`	0x0	**16** or **32**(1)	Write	Every data `write` to this address indicates this bitstream was sent to the IP core. Performing a `read` on this address returns all 0's.
`PR_CSR`	0x1	**16** or **32**(1)	Read/Write	Controls and status registers.

(1) Depending on the Avalon Memory Mapped data bus width.

Table 7: PR_CSR Control and Status Registers

Bit Offset	Description
0	Read and write control register for pr_start signal. Refer to "Input/output Port List" section for more details on pr_start signal. `pr_start = PR_CSR[0]` `PR_CSR[0]` is de-asserted to value "0" by the IP core automatically one clock cycle after it is asserted to streamline the flow so you do not need to manually assert and de-assert this register to control `pr_start` signal.
1	Read and write control register for `double_pr` signal. `double_pr = PR_CSR[1]`
2-4	Read only status register for `status[2:0]` signal. `PR_CSR[4:2] = status[2:0]`
5-15 or **5-31**(1)	Reserved

Related Information

- **Avalon Memory Map Slave Interface Read and Write Transfer Timing** on page 17
- **Partial Reconfiguration IP Core Ports** on page 8

FPGA Control Block Interface

When you instantiate the PR IP core, you can choose to use it as either an internal host or external host.

If it is used as an internal host, the PR IP core auto instantiates the corresponding device `crcblock` and `prblock` WYSIWYG atom primitive. If PR is used as external host (placed in another FPGA or CPLD), the PR IP core provides the `crcblock` and `prblock` interface ports so you can connect the host to the dedicated PR pins and CRC_ERROR pin on the target FPGA being partially reconfigured.

Note: You may need to instantiate the PR IP core as an external host to share the `crcblock` and `prblock` interface even though the PR IP core is located inside the FPGA being partially reconfigured. For example; your design uses another piece of IP, which instantiates its own `crcblock` WYSIWYG atom primitive to unload the Error Message Register (EMR) whenever `CRC_ERROR` is detected. Your design must share the `crcblock` interface signals between the PR IP core and secondary IP, otherwise the Quartus II compilation will fail because more than one `crcblock` is not allowed.

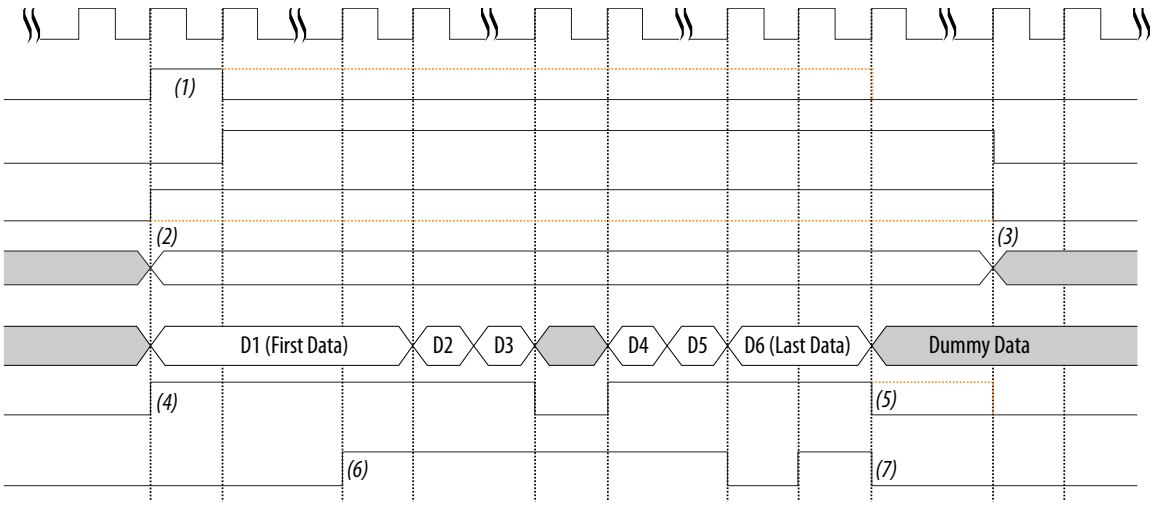
You can also instantiate the PR IP core as the external host so be able to add `crcblock` and `problock` WYSIWYG atom primitive instances to SignalTap II for debugging purposes.

Partial Reconfiguration IP Core Timing Specification

This timing diagram illustrates a successful Partial Reconfiguration IP core operation. Pass or fail can be determined with the `status[2:0]` output signal.

The PR operation is initated when you assert the `pr_start` signal. You can monitor the `status[]` or `freeze` signals to detect the end of the PR operation.

Figure 5: Partial Reconfiguration Timing



Note:

1. You must assert `pr_start` signal high for a minimum of one clock cycle to initiate PR and deassert `pr_start` before sending the last data.
2. `status[]` signal is reset when `pr_start` is asserted and changes during a PR operation if any error such as a `CRC_ERROR`, `PR_ERROR`, or bitstream incompatibility error is detected.
3. `status[]` signal changes after a PR operation if `CRC_ERROR` is detected and no error happens during the previous PR operation.
4. The `data_valid` signal is not required to be asserted at the same time as the `pr_start`. You can provied the `data[]` and assert `data_valid` when appropriate.
5. You can either drive the `data_valid` signal low after sending the last data, or contiue to assert `data_valid` high with dummy data on `data[]` until the end of PR, when `freeze` is driven low or `status[]` is updated/
6. `data[]` is transferred only when `data_valid` and `data_ready` are asserted on the same cycle. Do not drive new data on the data bus, when both `data_valid` and `data_ready` have not been asserted high.
7. The `data_ready` signal is driven low once the PR IP Core receives the last data.

The `data[]`, `data_valid`, and `data_ready` signals comply with the Avalon-ST specification for Data Transfer with Backpressure. The PR IP Core acts as a sink, with `readLatency = 0`. For more information, refer to the *Avalon Interface Specifications*.

If your `CDRATIO` is not 1, or your `data[]` width is x32 for series-V devices, the `data_ready` alternates between high and low for one or more clock cycles. The PR IP Core requires additional clock cycles to process the data recieved. You should hold `data[]` when `data_ready` is low.

Important: The `PR_CLK` signal has a different nominal maximum frequency for each device. Most Stratix V devices have a nominal maximum frequency of at least 62.5 MHz.

Related Information

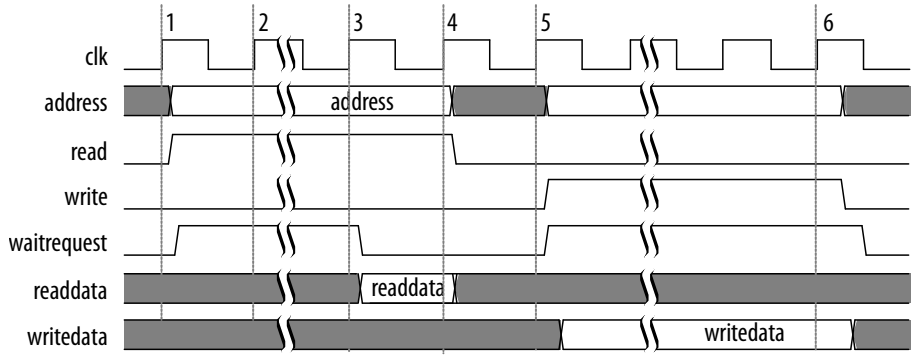
link/nik1412467993397/nik1412467975241
Avalon Interface Specifications for data transfer with backpressure

Avalon Memory Map Slave Interface Read and Write Transfer Timing

The Avalon-MM interface supports read and write transfers with a slave-controlled `waitrequest`. The slave can stall the interconnect for as many cycles as required by asserting the `waitrequest` signal. If a slave uses `waitrequest` for either read or write transfers, it must use `waitrequest` for both.

A slave typically receives `address`, `read` or `write`, and `writedata` after the rising edge of the clock. A slave asserts `waitrequest` before the rising clock edge to hold off transfers. When the slave asserts `waitrequest`, the transfer is delayed. And, the address and control signals are held constant. Transfers complete on the rising edge of the first `clk` after the slave port deasserts `waitrequest`.

Figure 6: Read and Write Transfers for Avalon-MM Slave Interface



The numbers in this timing diagram, mark the following transitions:

1. `address` and `read` are asserted after the rising edge of `clk`. `waitrequest` is asserted stalling the transfer.
2. `waitrequest` is sampled. Because `waitrequest` is asserted, the cycle becomes a wait-state. `address`, `read`, and `write` remain constant.
3. The slave presents valid `readdata` and deasserts `waitrequest`.
4. `readdata` and deasserted `waitrequest` are sampled, completing the transfer.
5. `address`, `writedata`, and `write` signals are asserted. The slave responds by asserting `waitrequest` stalling the transfer.
6. The slave captures `writedata` and deasserts `waitrequest` ending the transfer.

Related Information

Avalon Memory Mapped Interfaces
For more information on read and write transfers with Avalon Memory Mapped Interfaces

Control Block Interface Controller

This controller handles all the handshaking signals of the `prblock` WYSIWYG atom primitive and it will monitor `CRC_ERROR` signals of `crcblock` WYSIWYG atom primitives throughout the PR event (before, during, and after PR) to detect any CRAM error.

The controller receives data and sends it to the `prblock` WYSIWYG atom primitive during a PR event with the clock-to-data ratio (CDRATIO) you specify when you instantiate the Partial Reconfiguration IP core.

Table 8: CDRATIO for PR Compression and/or Encryption

The following table lists the range of values for the clock-to-data ratio for bitstreams that are compressed, encrypted, both, or neither.

Compressed	Encrypted	Clock-to-Data Ratio
No	No	x1
No	Yes	x2
Yes	No	x4
Yes	Yes	x4

Related Information

Instantiating the Partial Reconfiguration IP Core in the Quartus II IP Catalog on page 3

Freeze and Unfreeze Controls

The Freeze/Unfreeze controller logic of the Partial Reconfiguration IP core provides a signal to the user specified PR regions, freezing all input signals during PR operation to avoid current contention.

You begin a PR event by asserting the `pr_start` input port to the PR IP core. The Freeze/Unfreeeze Controller then asserts the output freeze signal high and sends it to the PR region you specified. All the input signals coming into that PR region are then pulled high, preventing current contention in the device. The PR IP core does not differentiate between multiple PR regions, so you may need to create custom freeze logic if you are freezing individual PR regions.

Once the PR operation is completed, this controller asserts the output freeze signal low. All the input signals coming into the specified PR region are released and the PR region is ready for normal operation.

Related Information

- **Partial Reconfiguration IP Core Timing Specification** on page 16
- **Partial Reconfiguration IP Core Ports** on page 8

Data Source Controller

This controller handles the source of PR data, either from JTAG or standard data interface.

The JTAG interface takes precedence over the standard PR data interface. For example, whenever JTAG is engaged through command from Quartus II Programmer tool, the PR data is sourced from the JTAG interface rather than the PR data interface.

Standard Partial Reconfiguration Data Interface

The PR data interface provides you with selectable input data width; x1, x2, x4, x8, x16, and x32. It can be connected to ASMI_PARALLEL as well as the Avalon interface to obtain PR data from on-chip RAM, external flash device, or PR over PCIe.

If the input data width is other than x16, the PR IP core includes a data upsize or downsize module so that the data output to the Data Source Controller is always x16.

JTAG Debug Mode for Partial Reconfiguration

The JTAG debug mode allows you to configure partial reconfiguration bitstream through the JTAG interface. Use this feature to debug PR bitstream and eventually helping you in your PR design prototyping. This feature is available for internal and external host.

During JTAG debug operation, the JTAG command sent from the Quartus II Programmer ignores and overrides most of the Partial Reconfiguration IP core interface signals (clk, pr_start, double_pr, data[], data_valid, and data_read).

Note: The TCK is the main clock source for PR IP core during this operation.

You can view the status of Partial Reconfiguration operation in the messages box and the Progress bar in the Quartus II Programmer. The PR_DONE, PR_ERROR, and CRC_ERROR signals will be monitored during PR operation and reported in the Messages box at the end of the operation.

The Quartus II Programmer can detect the number of PR_DONE instruction(s) in plain or compressed PR bitstream and, therefore, can handle single or double PR cycle accordingly. However, only single PR cycle is supported for encrypted Partial Reconfiguration bitstream in JTAG debug mode (provided that the specified device is configured with the encrypted base bitstream which contains the PR IP core in the design).

Note: Configuring an incompatible PR bitstream to the specified device may corrupt your design, including the routing path and the PR IP core placed in the static region. When this issue occurs, the PR IP core stays in an undefined state, and the Quartus II Programmer is unable to reset the IP core. As a result, the Quartus II Programmer generates the following error when you try to configure a new PR bitstream:

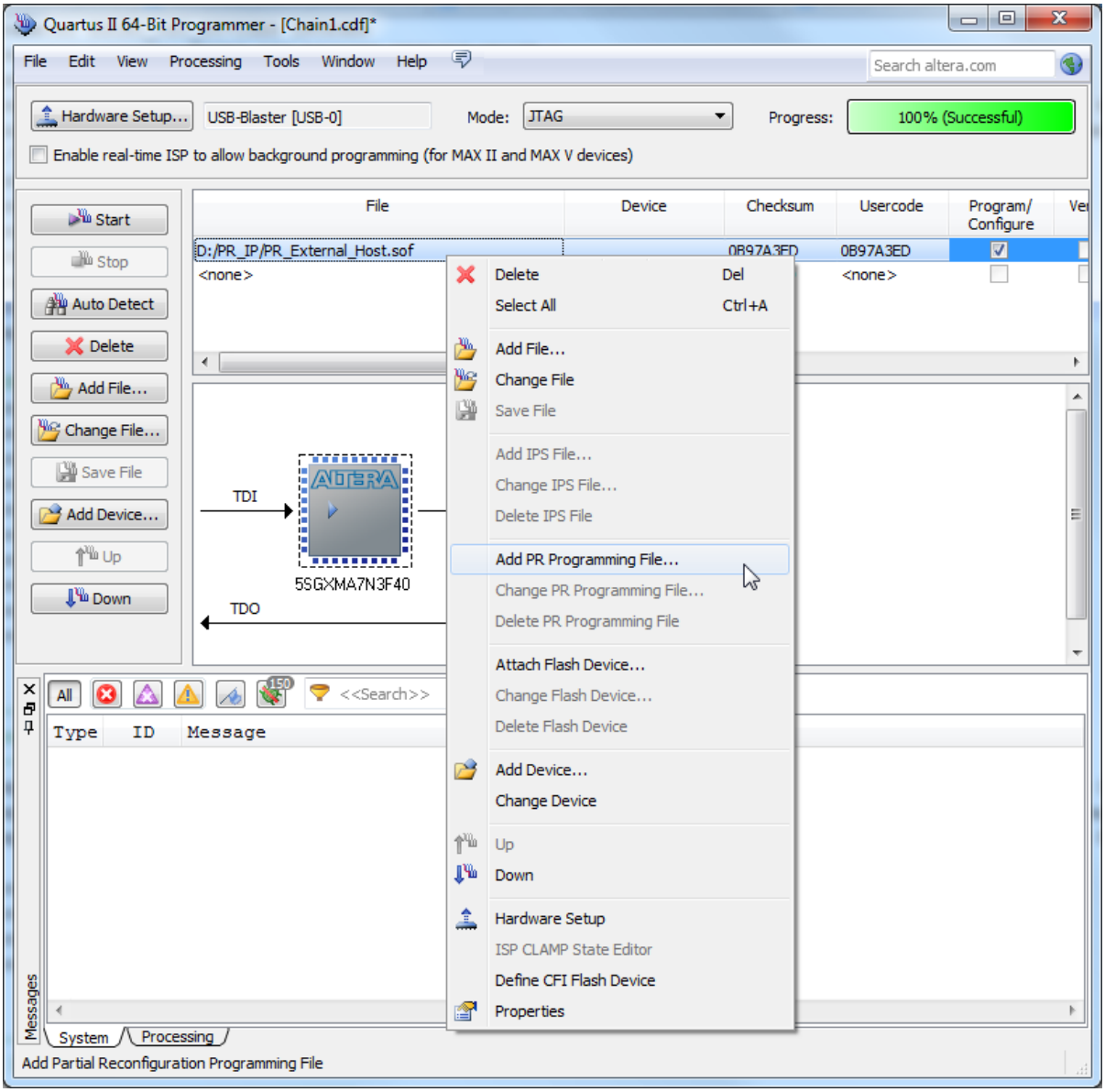
```
Error (12897): Partial Reconfiguration status: Can't reset the PR megafunction.
This issue occurred because the design was corrupted by an incompatible PR
bitstream in the previous PR operation. You must reconfigure the device with a good
design.
```

Configuring Partial Reconfiguration Bitstream in JTAG Debug Mode

To configure the Partial Reconfiguration bitstream in JTAG debug mode, follow these steps:

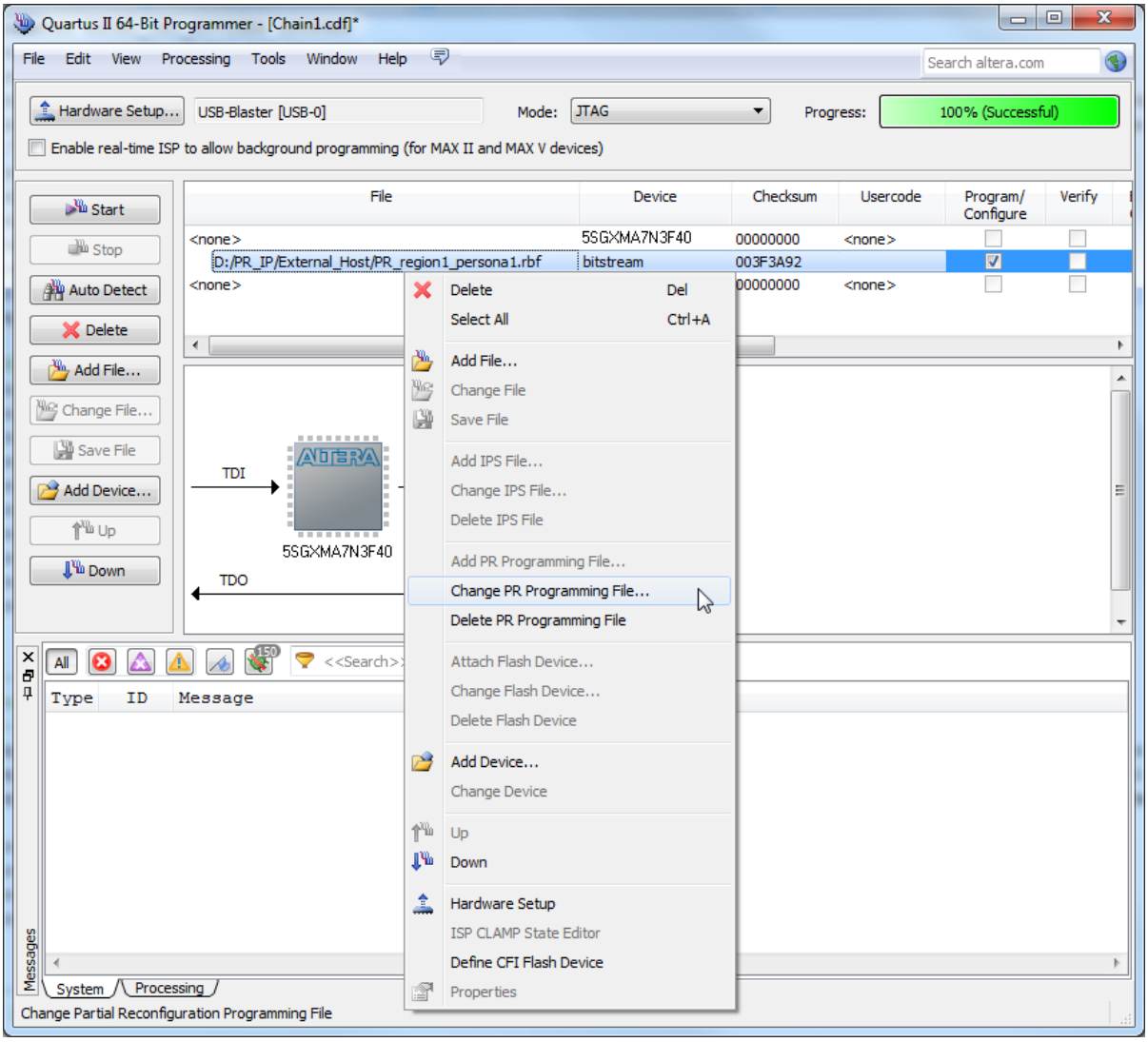
1. In the Quartus II Programmer GUI, right click on a highlighted base bitstream (in **.sof**) and then click **Add PR Programming File** to add the PR bitstream (**.rbf**).

Figure 7: Adding PR Programming File



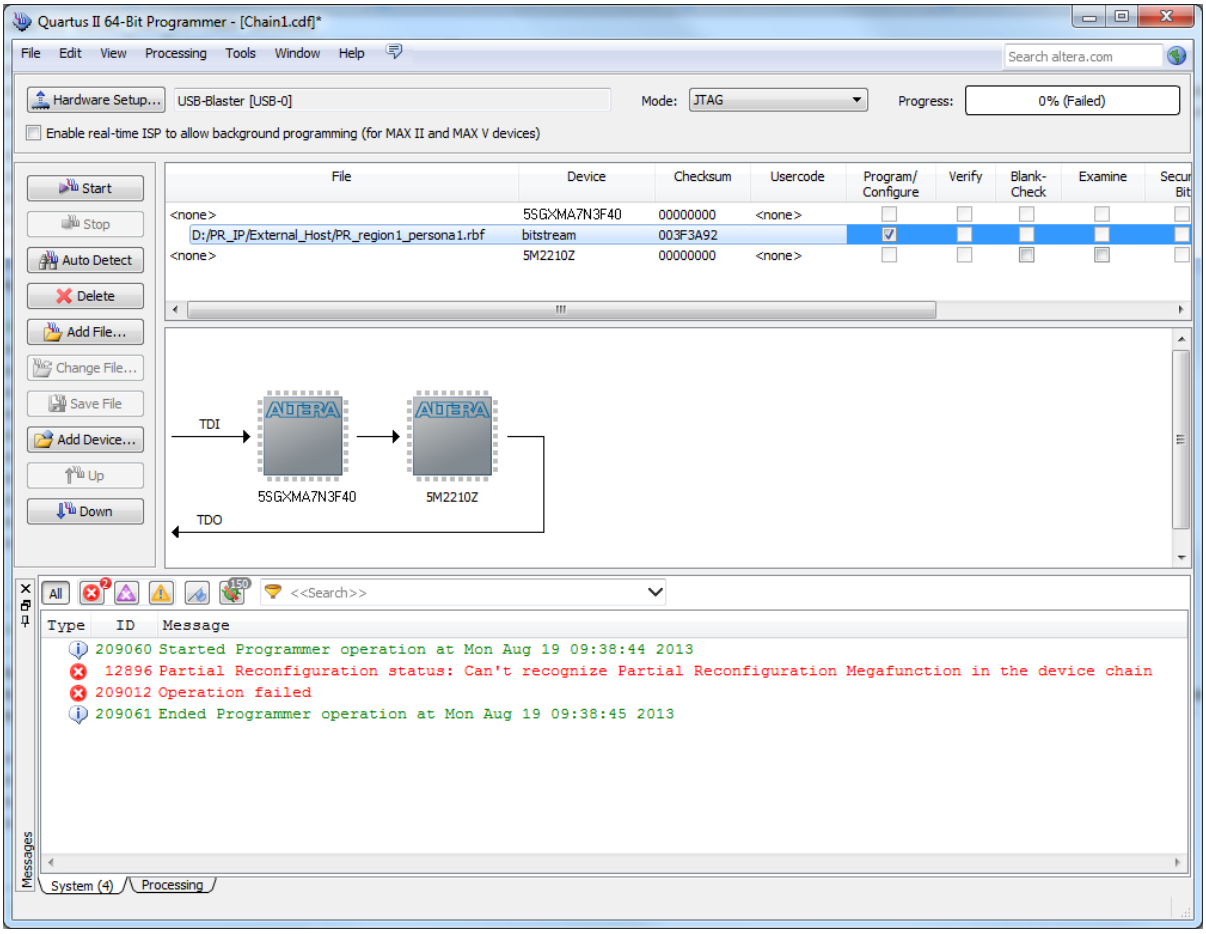
2. After adding thePR bitstream, you can change or delete the Partial Reconfiguration programming file by clicking **Change PR Programming File** or **Delete PR Programming File**.

Figure 8: Change PR Programming File or Delete PR Programming File



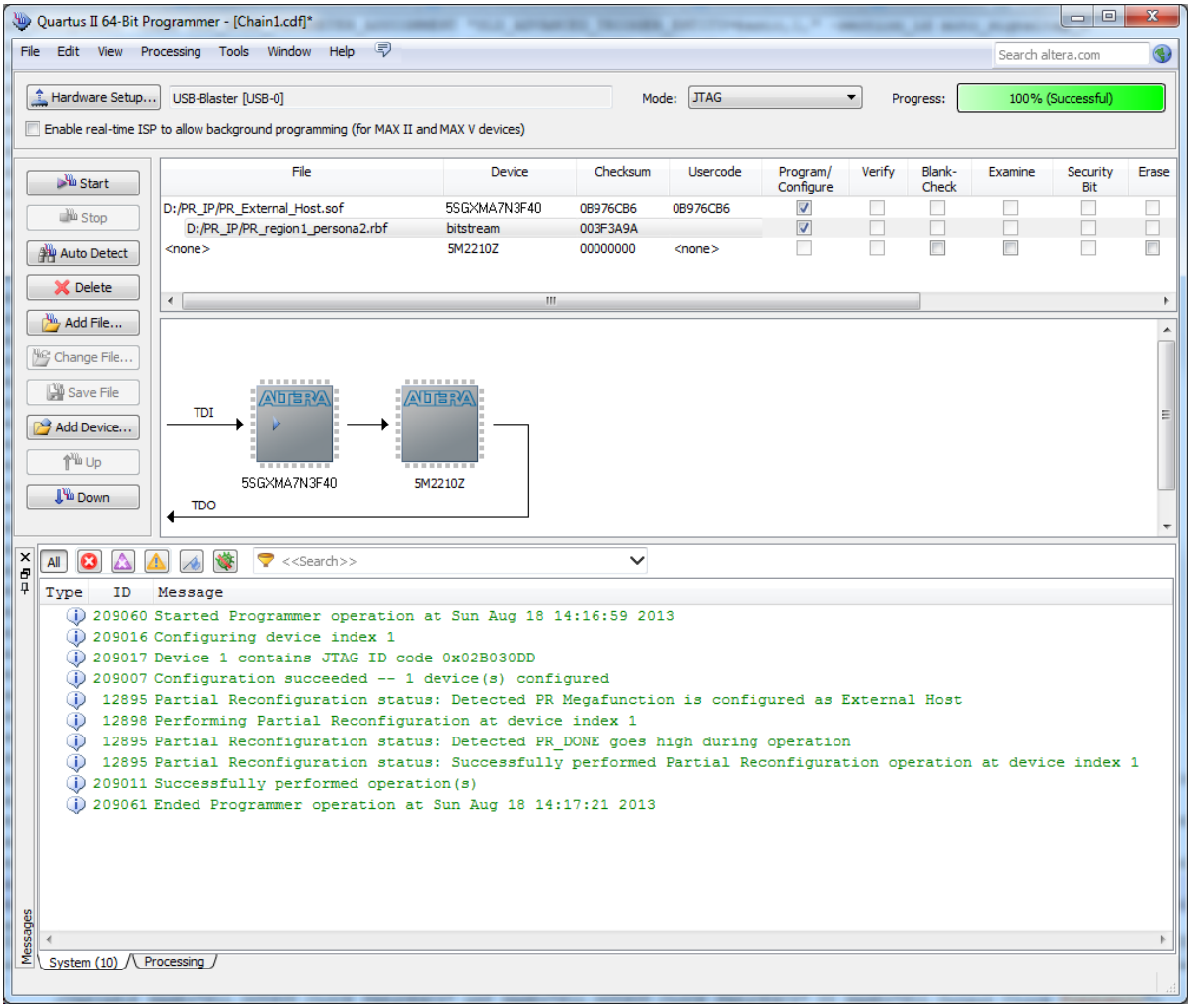
3. Click **Start** to configure the PR bitstream. The Quartus II Programmer generates an error message if the specified device does not contain the PR IP core in the design (you must instantiate the Partial Reconfiguration IP core in your design to use the JTAG debug mode).

Figure 9: Starting PR Bitstream Configuration



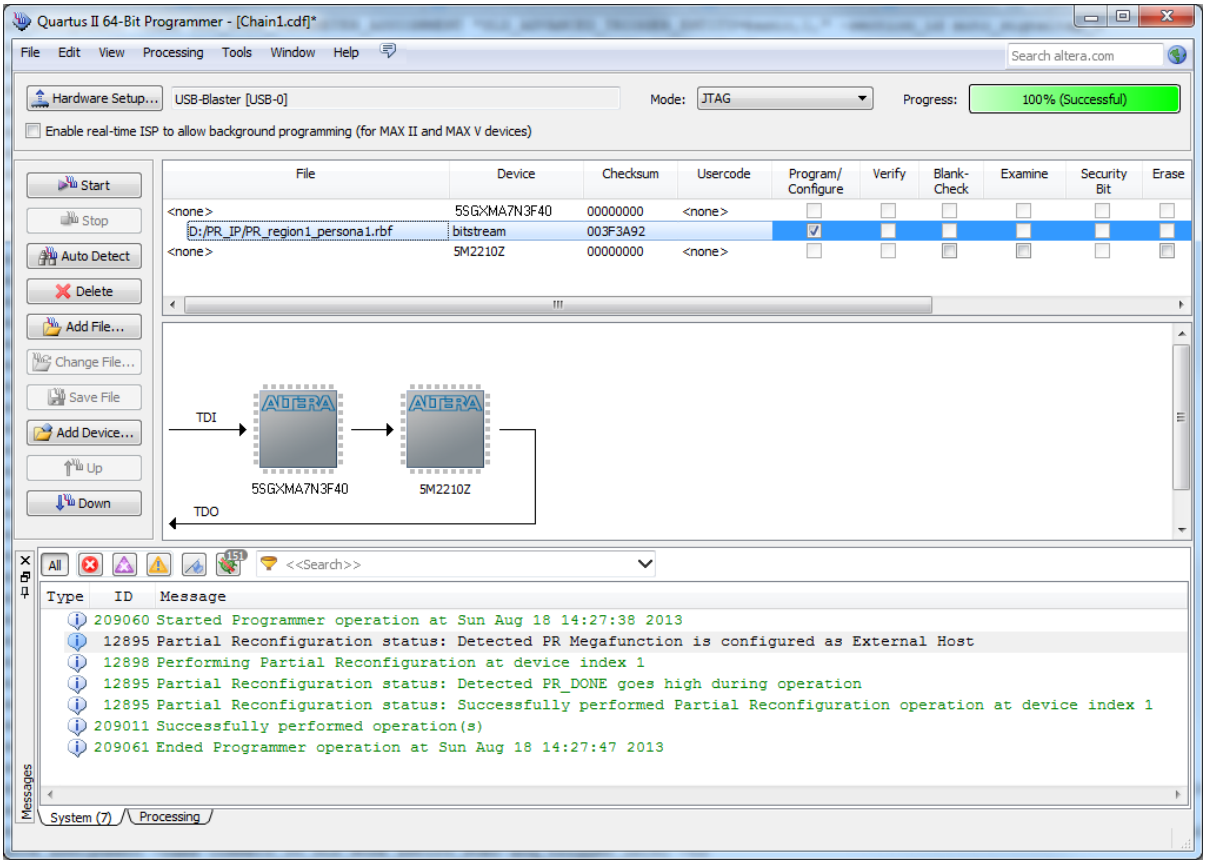
4. Configure the valid **.rbf** in JTAG debug mode with the Quartus II Programmer.

Figure 10: Configuring Valid .rbf



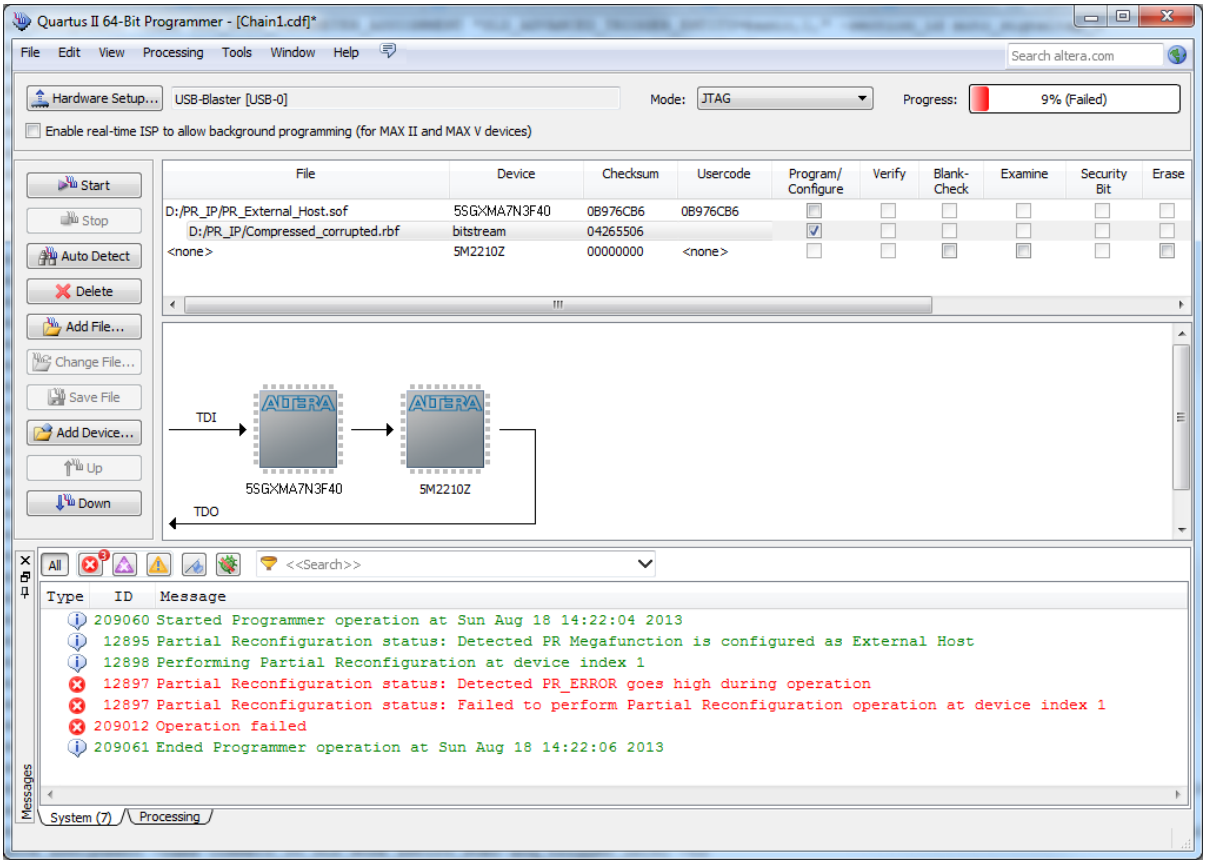
5. The JTAG debug mode is also supported if the PR IP core is pre-programmed on the specified device.

Figure 11: Partial Reconfiguration IP Core Successfully Pre-programmed



6. The Quartus II Programmer reports error when you try to configure the corrupted **.rbf** in JTAG debug mode.

Figure 12: Configuring Corrupted .rbf



Sample Freeze Wrapper for Multiple PR Regions

The following Verilog HDL pseudocode shows an example of how to create a simple freeze wrapper for two PR regions.

The same method can be applied to any number of PR regions.

```
// pseudocode for a simple freeze wrapper for two PR regions
module design_top (…);
  // user selects region A or B to be PR'ed
  input pr_region_A_or_B;
  input pr_start;

  assign freeze_A_w = pr_region_A_or_B ? pr_freeze_w : 1'b0;
  assign freeze_B_w = pr_region_A_or_B ? 1'b0 : pr_freeze_w;

  // freeze output of PR IP core
  alt_pr_sv my_alt_pr (
    .freeze (pr_freeze_w), // always stays low until user asserts pr_start
   .pr_start (pr_start),
    …
   );
   // Freeze wrapper for input signals of single PR region A
   // Follow existing recommendations in the PR user guide for the details
  freeze_region_A
```

```
  my_freeze_region_A (
    .freeze(freeze_A_w),
     …
  );

  // Freeze wrapper for input signals of single PR region B
  // Follow existing recommendations in the PR user guide for the details
  freeze_region_B my_freeze_region _B (
   .freeze(freeze_B_w),
    …
  );
endmodule
```

Related Information

Design Planning for Partial Reconfiguration
For more information on creating a freeze wrapper for partial reconfiguration.

Sample PR IP Core as an External Host on the Same Device

There are occasions where you should instantiate the PR IP core as external host on the same device.

1. To monitor the `prblock` and `crcblock` WYSIWYG interface signals using the SignalTap II tool or to probe these signals by routing them to any GPIO.
2. To share the `prblock` and `crcblock` WYSIWYG interface signals with another IP. For example, using the Fault Injection IP or a user controller to unload the Error Message Register (EMR) when a `CRC_ERROR` is asserted.

The following Verilog HDL pseudocode shows an example of how to instantiate the PR IP core as external host on the same device.

```
// pseudocode for instantiating the PR IP core as ExternalHost
// on the same device
module design_top (…);
  // PR IP core instantiated as External Host
  alt_pr_sv my_alt_pr (
      .pr_request_pin (pr_request_w),
      .pr_ready_pin (pr_ready_w),
      .pr_done_pin (pr_done_w),
      .pr_error_pin (pr_error_w),
      .pr_clk_pin (pr_clk_w),
      .pr_data_pin (pr_data_w),
      .crc_error_pin (crc_error_w),
      ...
  );

  // Stratix V prblock WYSIWYG
  stratixv_prblock my_prblock (
      .clk (pr_clk_w),
      .corectl(1'b1),    // note that this design still PR from core
      .prrequest (pr_request_w),
      .data (pr_data_w),
      .error (pr_error_w),
      .ready (pr_ready_w),
      .done (pr_done_w)
  );
  // Stratix V crcblock WYSIWYG
  stratixv_crcblock my_crcblock (
      .crcerror(crc_error_w),
      …
  );
endmodule
```

Related Information

Design Planning for Partial Reconfiguration
For more information on instantiating the an external host for partial reconfiguration.

Revision History

Date	Version	Changes
May 2015	2015.05.04	Revised the following topics: • Partial Reconfiguration IP Core Parameters—added new parameters for device family support • Partial Reconfiguration IP Core Ports—added new port options • Partial Reconfiguration IP Core Timing Specification —revised the timing diagram
January 2015	2015.01.29	Minor error corrections.
August 2014	2014.08.20	• Added Avalon Memory Map slave interface • Updated Ports and Parameters to support Avalon Memory Map slave interface • Added Bitstream compatibility checking • Added sample pseudo-code for creating a freeze wrapper for multiple PR regions and creating an external host on the same device.
November 2013	2013.11.04	Initial release